Sales Order Creation using Web Dynpro ABAP

**Applies to:**
SAP NetWeaver 2004s, Web Dynpro ABAP

**Summary**
This step by step exercise helps to create a sales order application using Web Dynpro ABAP.

**Author:** Kathirvel Balakrishnan  
**Company:** Wipro Technologies  
**Created on:** 07 February 2007

**Author Bio**
Kathirvel Balakrishnan is working as a ABAP Consultant for Wipro Technologies. His areas of interests are ABAP (Object, WebDynpro), Java, VBA and PHP Programming.
## Table of Contents

- Applies to: ................................................................................................................................. 1
- Summary ....................................................................................................................................... 1
- Author Bio ..................................................................................................................................... 1
- Prerequisite .................................................................................................................................. 3
- Sample Screen Shots .................................................................................................................. 3
- Step 1 – Create a Web Dynpro Component .................................................................................. 4
- Step 2 – Create nodes in the Component Controller ..................................................................... 4
- Step 3 – Create a methods to create sales order ........................................................................... 5
- Step 4 – Creating Views ................................................................................................................. 8
- Step 5 - Assigning Views to Window ............................................................................................ 9
- Step 6 – Create WebDynpro Application ..................................................................................... 10
- Related Content ............................................................................................................................ 11
- Disclaimer and Liability Notice ...................................................................................................... 12
Prerequisite
This example requires basic knowledge about ABAP.

Sample Screen Shots
The Web Dynpro ABAP Sales Order creation application,

Sales Order created in SAP R/3 System,

Display Standard Order 30000234: Overview
Step 1 – Create a Web Dynpro Component

Create a WebDynpro component with the window name SO_CREATE.

<table>
<thead>
<tr>
<th>Web Dynpro Component</th>
<th>ZKB_SO_CREATE</th>
<th>Active</th>
</tr>
</thead>
<tbody>
<tr>
<td>Description</td>
<td>SO Create</td>
<td></td>
</tr>
<tr>
<td>Assistance Class</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Step 2 – Create nodes in the Component Controller

Now double click the Component Controller and navigate to the Context Tab. Create four Nodes under that as shown below. In this demo application only a few parameters of the BAPI_SALESORDER_CREATEFROMDAT2 has been used. Incase real time scenarios one might require to use all the parameters. Hence there would be a need to create additional nodes in the Component Controller.

```
<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>ORDER_HEADER</td>
</tr>
<tr>
<td>Interface Node</td>
<td></td>
</tr>
<tr>
<td>Input Element (Ext)</td>
<td></td>
</tr>
<tr>
<td>Dictionary structure</td>
<td>BAPI_SODOC1</td>
</tr>
<tr>
<td>Cardinality</td>
<td>1:1</td>
</tr>
<tr>
<td>Selection</td>
<td>0:1</td>
</tr>
<tr>
<td>Initialization Lead Selection</td>
<td></td>
</tr>
<tr>
<td>Singleton</td>
<td></td>
</tr>
<tr>
<td>Supply Function</td>
<td></td>
</tr>
</tbody>
</table>

Node

<table>
<thead>
<tr>
<th>Property</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node</td>
<td>ORDER_ITEMS</td>
</tr>
<tr>
<td>Interface Node</td>
<td></td>
</tr>
<tr>
<td>Input Element (Ext)</td>
<td></td>
</tr>
<tr>
<td>Dictionary structure</td>
<td>BAPI_SODITM</td>
</tr>
<tr>
<td>Cardinality</td>
<td>1:1</td>
</tr>
<tr>
<td>Selection</td>
<td>0:1</td>
</tr>
<tr>
<td>Initialization Lead Selection</td>
<td></td>
</tr>
<tr>
<td>Singleton</td>
<td></td>
</tr>
<tr>
<td>Supply Function</td>
<td></td>
</tr>
</tbody>
</table>
```
Step 3 – Create a methods to create sales order

The source code for the method is given below,

```
METHOD create_sales_order .
```

* Declaration for Order Header Node

```plaintext
DATA: node_order_header TYPE REF TO if_wd_context_element,
     elem_order_header TYPE REF TO if_wd_context_element,
     stru_order_header TYPE if_componentcontroller=>element_order_header .
```

* Declaration for Order Items Node

```plaintext
DATA: node_order_items TYPE REF TO if_wd_context_element,
     elem_order_items TYPE REF TO if_wd_context_element,
     stru_order_items TYPE if_componentcontroller=>element_order_items .
```

* Declaration for Order Partners Node

```plaintext
DATA: node_order_partners TYPE REF TO if_wd_context_element,
```
elem_order_partners  TYPE REF TO if_wd_context_element,
stru_order_partners  TYPE if_componentcontroller=>element_order_partners
.

* Declaration for Order Conditions Node
DATA: node_order_conditions TYPE REF TO if_wd_context_node,
  elem_order_conditions TYPE REF TO if_wd_context_element,
  stru_order_conditions TYPE if_componentcontroller=>element_order_conditions.

* Read Context Details
* <ORDER_HEADER> via lead selection
  node_order_header = wd_context->get_child_node(  
    name = if_componentcontroller=>wdctx_order_header ).
  elem_order_header = node_order_header->get_element(  ).
  elem_order_header->get_static_attributes(  
    IMPORTING
    static_attributes = stru_order_header ).

* <ORDER_ITEMS> via lead selection
  node_order_items = wd_context->get_child_node(  
    name = if_componentcontroller=>wdctx_order_items ).
  elem_order_items = node_order_items->get_element(  ).
  elem_order_items->get_static_attributes(  
    IMPORTING
    static_attributes = stru_order_items ).

* <ORDER_PARTNERS> via lead selection
  node_order_partners = wd_context->get_child_node(  
    name = if_componentcontroller=>wdctx_order_partners ).
  elem_order_partners = node_order_partners->get_element(  ).
  elem_order_partners->get_static_attributes(  
    IMPORTING
    static_attributes = stru_order_partners ).

* <ORDERCONDITIONS> via lead selection
  node_order_conditions = wd_context->get_child_node(  
    name = if_componentcontroller=>wdctx_order_conditions ).
  elem_order_conditions = node_order_conditions->get_element(  ).
  elem_order_conditions->get_static_attributes(  
    IMPORTING
    static_attributes = stru_order_conditions ).

DATA: v_sales_doc TYPE bapivbeln-vbeln.

DATA: w_order_header_in     TYPE bapisdhd1,
  i_order_partners      TYPE TABLE OF bapiparnr,
  w_order_partners      TYPE bapiparnr,
  i_return              TYPE TABLE OF bapiret2,
  w_return              TYPE bapiret2,
  i_order_items_in      TYPE TABLE OF bapisditm,
  w_order_items_in      TYPE bapisditm,
  i_order_conditions_in TYPE TABLE OF bapicond,
  w_order_conditions_in TYPE bapicond.

* Order Header Details
CLEAR w_order_header_in.
w_order_header_in-doc_type = stru_order_header-doc_type.
w_order_header_in-sales_org = stru_order_header-sales_org.
w_order_header_in-distr_chan = stru_order_header-distr_chan.
w_order_header_in-division = stru_order_header-division.
w_order_header_in-reg_date_h = stru_order_header-reg_date_h.
w_order_header_in-sales_dist = stru_order_header-sales_dist.

* Order Item Details
CLEAR: i_order_items_in, w_order_items_in.
w_order_items_in-material = stru_order_items-material.
w_order_items_in-salqtynum = stru_order_items-salqtynum.
CALL FUNCTION 'CONVERSION_EXIT_ALPHA_INPUT'
EXPORTING
  input  = w_order_items_in-material
IMPORTING
  output = w_order_items_in-material.
APPEND w_order_items_in TO i_order_items_in.

* Order Partner Details
CLEAR: i_order_partners, w_order_partners.
w_order_partners-partn_role = stru_order_partners-partn_role.
w_order_partners-partn_numb = stru_order_partners-partn_numb.
CALL FUNCTION 'CONVERSION_EXIT_ALPHA_INPUT'
EXPORTING
  input  = w_order_partners-partn_numb
IMPORTING
  output = w_order_partners-partn_numb.
APPEND w_order_partners TO i_order_partners.

* Order Condition Details
CLEAR: i_order_conditions_in, w_order_conditions_in.
w_order_conditions_in-itm_number = stru_order_conditions-itm_number.
w_order_conditions_in-cond_type  = stru_order_conditions-cond_type.
w_order_conditions_in-cond_value = stru_order_conditions-cond_value.
APPEND w_order_conditions_in TO i_order_conditions_in.

* Create Sales Order BAPI Call
CALL FUNCTION 'BAPI_SALESORDER_CREATEFROMDAT2'
EXPORTING
  order_header_in     = w_order_header_in
IMPORTING
  salesdocument       = v_sales_doc
TABLES
  return              = i_return
  order_items_in      = i_order_items_in
  order_partners      = i_order_partners
  order_conditions_in = i_order_conditions_in.

* Get message manager
DATA: l_current_controller TYPE REF TO if_wd_controller,
    l_message_manager  TYPE REF TO if_wd_message_manager.

l_current_controller ?= wd_this->wd_get_api( ).

CALL METHOD l_current_controller->get_message_manager
RECEIVING
  message_manager = l_message_manager.
DATA: v_message_text TYPE string.

* When Sales Order is created commit the data
  IF NOT v_sales_doc IS INITIAL.
    CALL FUNCTION 'BAPI_TRANSACTION_COMMIT'
      EXPORTING
        wait = 'X'.
    CONCATENATE 'Sales Document' v_sales_doc 'has been created.'
    INTO v_message_text SEPARATED BY space.
  * Report Success message
    CALL METHOD l_message_manager->report_success
      EXPORTING
        message_text = v_message_text.
  ELSE.
    v_message_text = 'Error Creating Sales Order'.
  * Report Error message
    CALL METHOD l_message_manager->report_error_message
      EXPORTING
        message_text = v_message_text.
  ENDIF.
ENDMETHOD.

Step 4 – Creating Views

Create a view named MAIN_VIEW and map the all node from the component controller to it.

Now design the layout as shown below. Create 4 Groups one for each node and add a button named CREATE to it.
Create an ACTION named CREATE_SALES_ORD for the CREATE button.

The implementation for the ONACTIONCREATE_SALES_ORD method is given below,

METHOD onactioncreate_sales_ord.

* Here we call the component Controller Method
  * that will create the sales order
    wd_comp_controller->create_sales_order().

ENDMETHOD.

Step 5 - Assigning Views to Window
Now drag the MAIN_VIEW and drop it on the Window SO_CREATE.
Step 6 – Create WebDynpro Application

Create a Web Dynpro application and Test the program.
Related Content

Please include at least three references to SDN documents or web pages.


https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/7a89b067-0801-0010-8192-a9896a60938e


https://www.sdn.sap.com/irj/servlet/prt/portal/prtroot/docs/library/uuid/a282c952-0801-0010-1eb5-87953e036712
Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.